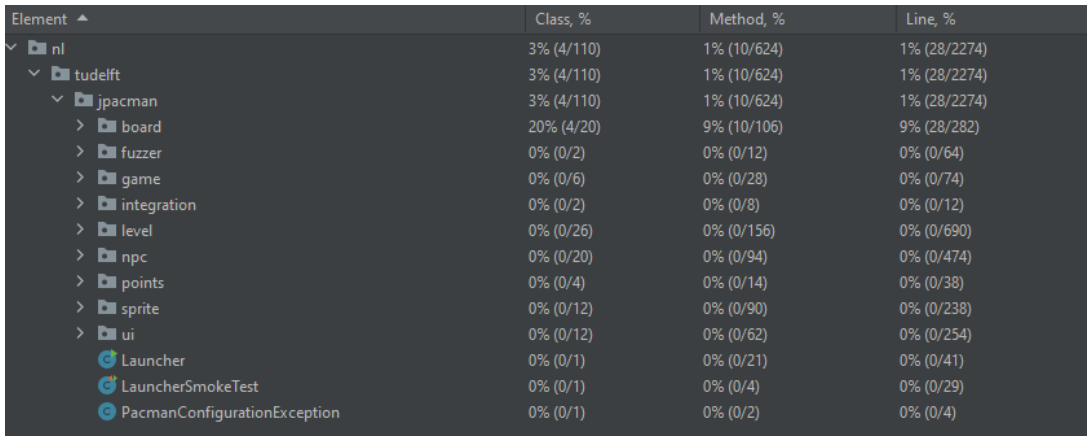
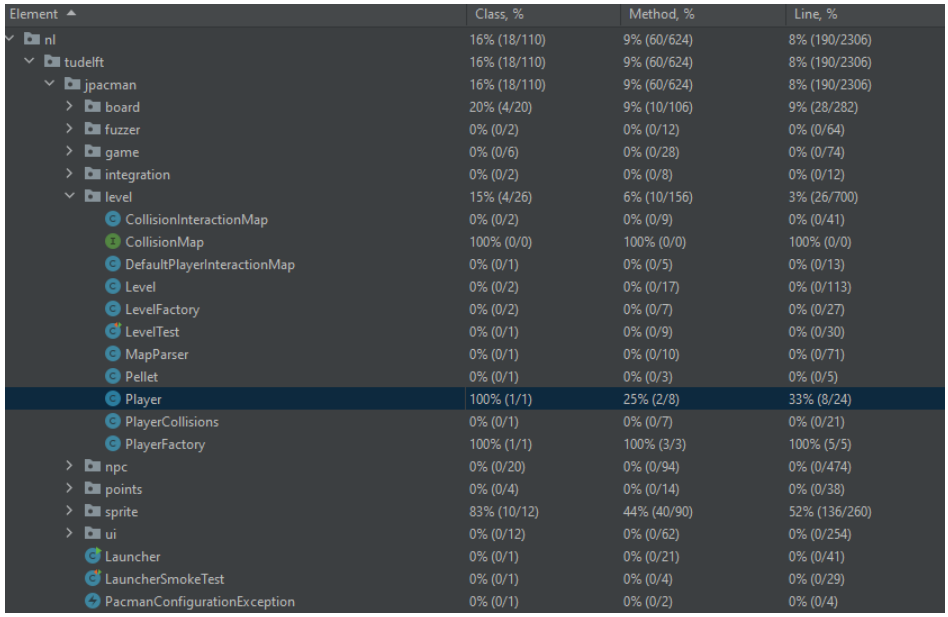
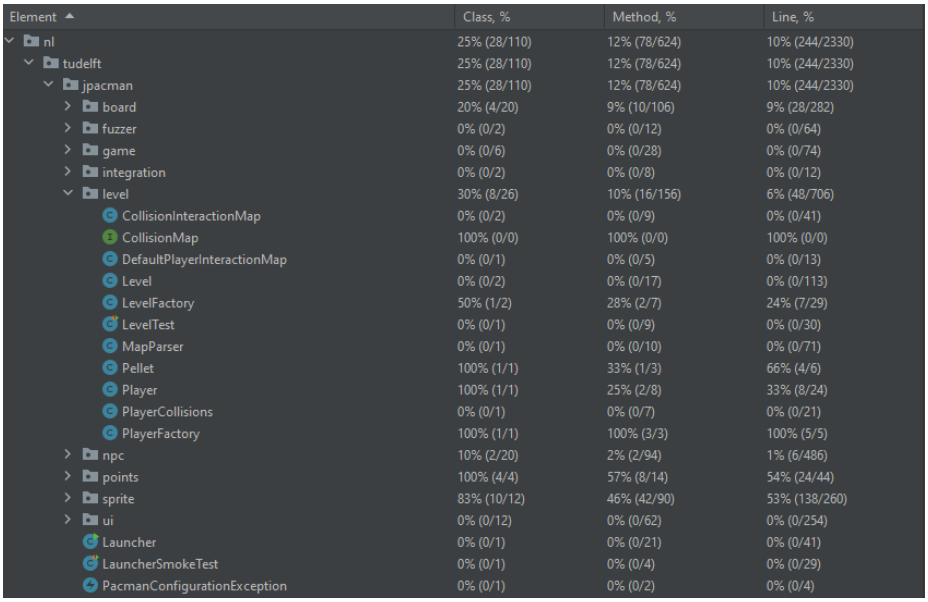
- Repo Link: https://github.com/carruthjhall/cs472-team6
- Task 1
    - Class, Method, Line
    - First Coverage Test
        - 

Element	Class, %	Method, %	Line, %
nl	3% (4/110)	1% (10/624)	1% (28/2274)
tudelft	3% (4/110)	1% (10/624)	1% (28/2274)
jpacman	3% (4/110)	1% (10/624)	1% (28/2274)
board	20% (4/20)	9% (10/106)	9% (28/282)
fuzzer	0% (0/2)	0% (0/12)	0% (0/64)
game	0% (0/6)	0% (0/28)	0% (0/74)
integration	0% (0/2)	0% (0/8)	0% (0/12)
level	0% (0/26)	0% (0/156)	0% (0/690)
npc	0% (0/20)	0% (0/94)	0% (0/474)
points	0% (0/4)	0% (0/14)	0% (0/38)
sprite	0% (0/12)	0% (0/90)	0% (0/238)
ui	0% (0/12)	0% (0/62)	0% (0/254)
Launcher	0% (0/1)	0% (0/21)	0% (0/41)
LauncherSmokeTest	0% (0/1)	0% (0/4)	0% (0/29)
PacmanConfigurationException	0% (0/1)	0% (0/2)	0% (0/4)

        - Coverage started at 3% Class, 1% Method, and 1% Line
    - Added playerTest
        - 

Element	Class, %	Method, %	Line, %
nl	16% (18/110)	9% (60/624)	8% (190/2306)
tudelft	16% (18/110)	9% (60/624)	8% (190/2306)
jpacman	16% (18/110)	9% (60/624)	8% (190/2306)
board	20% (4/20)	9% (10/106)	9% (28/282)
fuzzer	0% (0/2)	0% (0/12)	0% (0/64)
game	0% (0/6)	0% (0/28)	0% (0/74)
integration	0% (0/2)	0% (0/8)	0% (0/12)
level	15% (4/26)	6% (10/156)	3% (26/700)
CollisionInteractionMap	0% (0/2)	0% (0/9)	0% (0/41)
CollisionMap	100% (0/0)	100% (0/0)	100% (0/0)
DefaultPlayerInteractionMap	0% (0/1)	0% (0/5)	0% (0/13)
Level	0% (0/2)	0% (0/17)	0% (0/113)
LevelFactory	0% (0/2)	0% (0/7)	0% (0/27)
LevelTest	0% (0/1)	0% (0/9)	0% (0/30)
MapParser	0% (0/1)	0% (0/10)	0% (0/71)
Pellet	0% (0/1)	0% (0/3)	0% (0/5)
Player	100% (1/1)	25% (2/8)	33% (8/24)
PlayerCollisions	0% (0/1)	0% (0/7)	0% (0/21)
PlayerFactory	100% (1/1)	100% (3/3)	100% (5/5)
npc	0% (0/20)	0% (0/94)	0% (0/474)
points	0% (0/4)	0% (0/14)	0% (0/38)
sprite	83% (10/12)	44% (40/90)	52% (136/260)
ui	0% (0/12)	0% (0/62)	0% (0/254)
Launcher	0% (0/1)	0% (0/21)	0% (0/41)
LauncherSmokeTest	0% (0/1)	0% (0/4)	0% (0/29)
PacmanConfigurationException	0% (0/1)	0% (0/2)	0% (0/4)

        - After adding the playerTest, coverage jumped up to 16%-C, 9%-M, 8%-L
    - Added pelletTest - src/main/java/nl/tudelft/jpacman/level/LevelFactory.java/createPellet

- 

Element	Class, %	Method, %	Line, %
nl	25% (28/110)	12% (78/624)	10% (244/2330)
tudelft	25% (28/110)	12% (78/624)	10% (244/2330)
jpacman	25% (28/110)	12% (78/624)	10% (244/2330)
board	20% (4/20)	9% (10/106)	9% (28/282)
fuzzer	0% (0/2)	0% (0/12)	0% (0/64)
game	0% (0/6)	0% (0/28)	0% (0/74)
integration	0% (0/2)	0% (0/8)	0% (0/12)
level	30% (8/26)	10% (16/156)	6% (48/706)
CollisionInteractionMap	0% (0/2)	0% (0/9)	0% (0/41)
CollisionMap	100% (0/0)	100% (0/0)	100% (0/0)
DefaultPlayerInteractionMap	0% (0/1)	0% (0/5)	0% (0/13)
Level	0% (0/2)	0% (0/17)	0% (0/113)
LevelFactory	50% (1/2)	28% (2/7)	24% (7/29)
LevelTest	0% (0/1)	0% (0/9)	0% (0/30)
MapParser	0% (0/1)	0% (0/10)	0% (0/71)
Pellet	100% (1/1)	33% (1/3)	66% (4/6)
Player	100% (1/1)	25% (2/8)	33% (8/24)
PlayerCollisions	0% (0/1)	0% (0/7)	0% (0/21)
PlayerFactory	100% (1/1)	100% (3/3)	100% (5/5)
npc	10% (2/20)	2% (2/94)	1% (6/486)
points	100% (4/4)	57% (8/14)	54% (24/44)
sprite	83% (10/12)	46% (42/90)	53% (138/260)
ui	0% (0/12)	0% (0/62)	0% (0/254)
Launcher	0% (0/1)	0% (0/21)	0% (0/41)
LauncherSmokeTest	0% (0/1)	0% (0/4)	0% (0/29)
PacmanConfigurationException	0% (0/1)	0% (0/2)	0% (0/4)

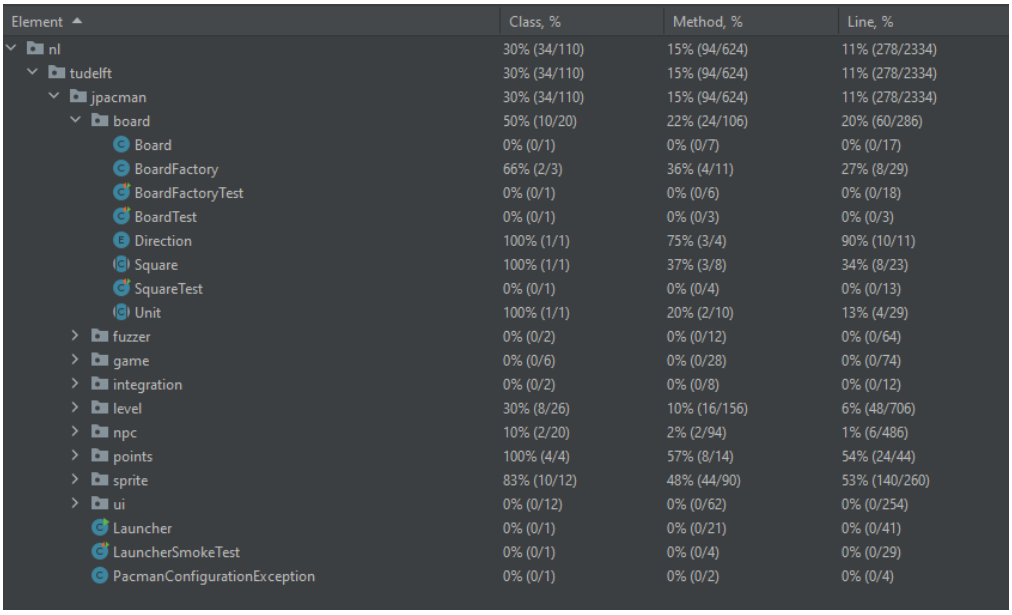
- ```
  package nl.tudelft.jpacman.level;

  import nl.tudelft.jpacman.npc.ghost.GhostFactory;
  import nl.tudelft.jpacman.points.PointCalculator;
  import nl.tudelft.jpacman.points.PointCalculatorLoader;
  import nl.tudelft.jpacman.sprite.PacManSprites;
  import static org.assertj.core.api.Assertions.assertThat;
  import org.junit.jupiter.api.Test;

  no usages  new *
  public class PelletTest {

      2 usages
      PacManSprites ps = new PacManSprites();
      1 usage
      GhostFactory ghost = new GhostFactory(ps);
      1 usage
      PointCalculator pc = new PointCalculatorLoader().load();
      1 usage
      LevelFactory lf = new LevelFactory(ps, ghost, pc);
      no usages  new *
      @Test
      void pelletTest(){
          assertThat(lf.createPellet()).isNotNull();
      }
  }
  ```

- Added createGroundTest - src/main/java/nl/tudelft/jpacman/board/BoardFactory.java/createGround
  ```



Element	Class, %	Method, %	Line, %
nl	30% (34/110)	15% (94/624)	11% (278/2334)
tudelft	30% (34/110)	15% (94/624)	11% (278/2334)
jpacman	30% (34/110)	15% (94/624)	11% (278/2334)
board	50% (10/20)	22% (24/106)	20% (60/286)
Board	0% (0/1)	0% (0/7)	0% (0/17)
BoardFactory	66% (2/3)	36% (4/11)	27% (8/29)
BoardFactoryTest	0% (0/1)	0% (0/6)	0% (0/18)
BoardTest	0% (0/1)	0% (0/3)	0% (0/3)
Direction	100% (1/1)	75% (3/4)	90% (10/11)
Square	100% (1/1)	37% (3/8)	34% (8/23)
SquareTest	0% (0/1)	0% (0/4)	0% (0/13)
Unit	100% (1/1)	20% (2/10)	13% (4/29)
fuzzer	0% (0/2)	0% (0/12)	0% (0/64)
game	0% (0/6)	0% (0/28)	0% (0/74)
integration	0% (0/2)	0% (0/8)	0% (0/12)
level	30% (8/26)	10% (16/156)	6% (48/706)
npc	10% (2/20)	2% (2/94)	1% (6/486)
points	100% (4/4)	57% (8/14)	54% (24/44)
sprite	83% (10/12)	48% (44/90)	53% (140/260)
ui	0% (0/12)	0% (0/62)	0% (0/254)
Launcher	0% (0/1)	0% (0/21)	0% (0/41)
LauncherSmokeTest	0% (0/1)	0% (0/4)	0% (0/29)
PacmanConfigurationException	0% (0/1)	0% (0/2)	0% (0/4)

-

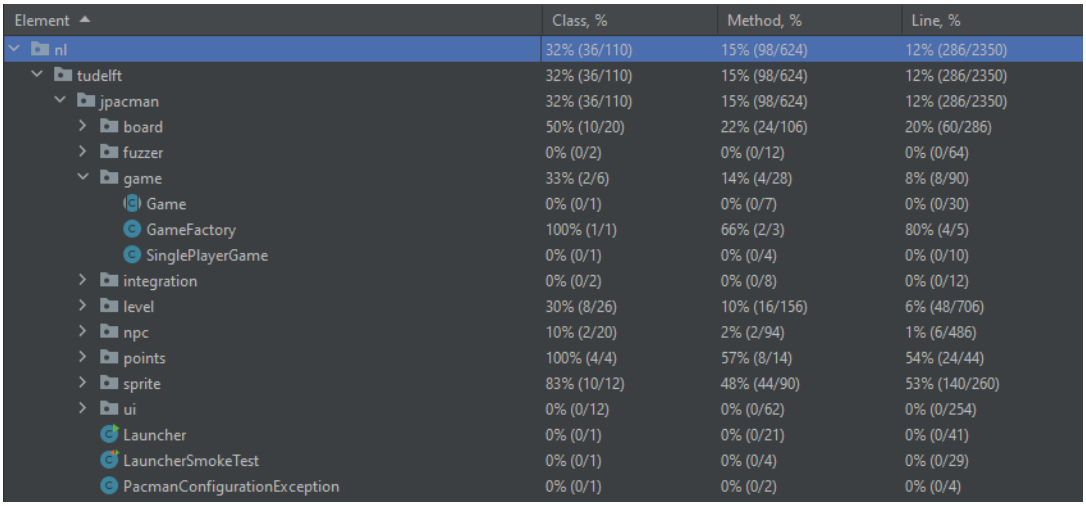
```java
package nl.tudelft.jpacman.board;
import nl.tudelft.jpacman.board.BoardFactory;
import nl.tudelft.jpacman.sprite.PacManSprites;
import org.junit.jupiter.api.Test;

import static org.assertj.core.api.Assertions.assertThat;

no usages  new *
public class createGroundTest {
    1 usage
    PacManSprites ps = new PacManSprites();
    1 usage
    BoardFactory b = new BoardFactory(ps);
    no usages  new *
    @Test
    void createGroundTest() { assertThat(b.createGround()).isNotNull(); }
}
```

-

- Added getPlayerFactoryTest - src/main/java/nl/tudelft/jpacman/game/GameFactory.java/getPlayerFactory

- 

Element	Class, %	Method, %	Line, %
nl	32% (36/110)	15% (98/624)	12% (286/2350)
tudelft	32% (36/110)	15% (98/624)	12% (286/2350)
jpacman	32% (36/110)	15% (98/624)	12% (286/2350)
board	50% (10/20)	22% (24/106)	20% (60/286)
fuzzer	0% (0/2)	0% (0/12)	0% (0/64)
game	33% (2/6)	14% (4/28)	8% (8/90)
Game	0% (0/1)	0% (0/7)	0% (0/30)
GameFactory	100% (1/1)	66% (2/3)	80% (4/5)
SinglePlayerGame	0% (0/1)	0% (0/4)	0% (0/10)
integration	0% (0/2)	0% (0/8)	0% (0/12)
level	30% (8/26)	10% (16/156)	6% (48/706)
npc	10% (2/20)	2% (2/94)	1% (6/486)
points	100% (4/4)	57% (8/14)	54% (24/44)
sprite	83% (10/12)	48% (44/90)	53% (140/260)
ui	0% (0/12)	0% (0/62)	0% (0/254)
Launcher	0% (0/1)	0% (0/21)	0% (0/41)
LauncherSmokeTest	0% (0/1)	0% (0/4)	0% (0/29)
PacmanConfigurationException	0% (0/1)	0% (0/2)	0% (0/4)

-

```java
package nl.tudelft.jpacman.game;
import static org.assertj.core.api.Assertions.assertThat;

import nl.tudelft.jpacman.level.PlayerFactory;
import nl.tudelft.jpacman.sprite.PacManSprites;
import nl.tudelft.jpacman.sprite.Sprite;
import org.junit.jupiter.api.Test;


no usages  new *
public class getPlayerFactoryTest {

    1 usage
    PacManSprites spr = new PacManSprites();
    1 usage
    PlayerFactory p = new PlayerFactory(spr);

    1 usage
    GameFactory g = new GameFactory(p);


    no usages  new *
    @Test
    void getPlayerFactoryTest(){
        assertThat(g.getPlayerFactory()).isNotNull();
    }
}
```

-
    -

- Task 3
    - The coverage results are not exactly the same as IntellJ. Intellj is saying there are 706 lines in “nl.tudelft.jpacman.level” for example, however Jacoco says there are 344 lines. I am assuming that Jacoco is omitting lines such as whitespace and/or lines that don’t necessarily require a check such as setting a local variable.
    - The source code visualization on Jacoco is significantly more helpful as since it shows the lines that are covered in green, and those not covered in red. This helps find exactly where we need to apply more coverage to.

- I preferJacoco's visualization since it lets you see the exact lines that need attention, however IntellJ's coverage window is more convenient since it is built into the workspace. That being said, Jacoco would be my go to in a real life situation.